INTERNET APPLICATIONS PROGRAMMING

PHP, MySQL, JavaScript & AJAX

Complete Full-Stack Web Development Course

Course Coverage:

PHP Server-Side Programming

MySQL Database Integration

Sessions and Cookies

User Authentication & Authorization

Form Handling and Validation

AJAX and Asynchronous JavaScript

REST APIs and Web Services

File Upload and Management

Security Best Practices

Email Integration

Payment Gateway Integration

Complete E-commerce Application

TABLE OF CONTENTS

1. MODULE 1: Introduction to Web Applications
2. MODULE 2: PHP Basics and Syntax
3. MODULE 3: PHP Variables, Arrays, and Functions
4. MODULE 4: PHP and HTML Integration
5. MODULE 5: Form Handling with PHP
6. MODULE 6: MySQL Database Fundamentals
7. MODULE 7: PHP-MySQL Connection
8. MODULE 8: CRUD Operations (Create, Read, Update, Delete)
9. MODULE 9: Sessions and Cookies
10. MODULE 10: User Authentication System
11. MODULE 11: File Upload and Management
12. MODULE 12: JavaScript for Web Applications
13. MODULE 13: AJAX - Asynchronous JavaScript and XML
14. MODULE 14: REST API Development
15. MODULE 15: Security Best Practices
16. MODULE 16: Email Integration with PHPMailer
17. MODULE 17: Payment Gateway Integration (M-Pesa)
18. MODULE 18: Complete Project - Student Portal
19. MODULE 19: Complete Project - E-commerce System
20. MODULE 20: Deployment and Hosting

MODULE 1: INTRODUCTION TO WEB APPLICATIONS

1.1 What is a Web Application?

A web application is a software program that:

- Runs on a web server
- Is accessed through a web browser
- Processes user requests and returns responses
- Stores data in a database
- Provides interactive functionality

1.2 Client-Server Architecture

CLIENT SIDE (Frontend):

• HTML - Structure and content

• CSS - Styling and layout

• JavaScript - Interactivity and behavior

SERVER SIDE (Backend):

• PHP - Server-side logic

• MySQL - Database storage

• Apache/Nginx - Web server

1.3 How Web Applications Work

1. User enters URL in browser

2. Browser sends HTTP request to web server

3. Server processes request with PHP

4. PHP queries database if needed

5. Server generates HTML response

6. Browser receives and displays HTML

7. JavaScript adds interactivity

1.4 Required Tools

Software Needed:

- XAMPP (Apache + MySQL + PHP) - Free download from apachefriends.org
- Text Editor - VS Code, Sublime Text, or PhpStorm
- Web Browser - Chrome, Firefox (with Developer Tools)
- Optional: MySQL Workbench for database design

1.5 XAMPP Installation and Setup

Installation Steps:

1. Download XAMPP from apachefriends.org

2. Run installer (choose default settings)

3. Install to C:\xampp (Windows) or /opt/lampp (Linux)

4. Launch XAMPP Control Panel

5. Start Apache and MySQL services

6. Test by visiting http://localhost in browser

7. You should see XAMPP dashboard

1.6 Creating Your First PHP File

Steps:

1. Navigate to C:\xampp\htdocs\

2. Create folder "myproject"

3. Create file "index.php" inside myproject

4. Add code below

5. Visit http://localhost/myproject/

```
<!DOCTYPE html>
<html>
<head>
    <title>My First PHP Page</title>
</head>
<body>
    <h1>Hello from PHP!</h1>

    <?php
```

```
        echo "<p>This is PHP code!</p>";
        echo "<p>Server date: " . date("Y-m-d") . "</p>";
        echo "<p>Server time: " . date("H:i:s") . "</p>";
    ?>

    <h2>PHP Information</h2>
    <?php
        echo "PHP Version: " . phpversion();
    ?>
</body>
</html>
```

MODULE 2: PHP BASICS AND SYNTAX

2.1 PHP Syntax Rules

Basic Rules:

- PHP code starts with <?php and ends with ?>
- Statements end with semicolon (;)
- Variables start with $ sign
- Case-sensitive for variable names
- Comments: // single line, /* */ multi-line

2.2 Variables and Data Types

```
<?php
// Variables
$name = "John Kamau";           // String
$age = 25;                      // Integer
$height = 5.9;                  // Float
$isStudent = true;              // Boolean
$fruits = array("Apple", "Mango", "Orange");  // Array

// Displaying variables
echo "Name: $name<br>";
echo "Age: $age<br>";
echo "Height: $height<br>";
echo "Is Student: $isStudent<br>";

// Variable concatenation
echo "Hello, my name is " . $name . " and I am " . $age . " years
old.";

// String interpolation
echo "<p>$name is $age years old.</p>";
?>
```

2.3 Operators

```
<?php
// Arithmetic Operators
$a = 10;
$b = 3;

echo "Addition: " . ($a + $b) . "<br>";        // 13
echo "Subtraction: " . ($a - $b) . "<br>";     // 7
echo "Multiplication: " . ($a * $b) . "<br>";  // 30
echo "Division: " . ($a / $b) . "<br>";        // 3.333...
echo "Modulus: " . ($a % $b) . "<br>";         // 1

// Comparison Operators
```

```
echo ($a > $b) ? "a is greater" : "b is greater";  // a is greater
echo ($a == $b) ? "Equal" : "Not Equal";           // Not Equal

// Increment/Decrement
$counter = 5;
$counter++;  // Now 6
$counter--;  // Back to 5

// String Operators
$firstName = "John";
$lastName = "Doe";
$fullName = $firstName . " " . $lastName;  // Concatenation
echo $fullName;
?>
```

2.4 Control Structures

```
<?php
// IF-ELSE Statement
$score = 85;

if ($score >= 90) {
    echo "Grade: A";
} elseif ($score >= 80) {
    echo "Grade: B";
} elseif ($score >= 70) {
    echo "Grade: C";
} elseif ($score >= 60) {
    echo "Grade: D";
} else {
    echo "Grade: F";
}

// SWITCH Statement
$day = "Monday";

switch ($day) {
    case "Monday":
        echo "Start of the week";
        break;
    case "Friday":
        echo "End of the week";
        break;
    case "Saturday":
    case "Sunday":
        echo "Weekend!";
        break;
    default:
        echo "Middle of the week";
}
```

```php
// FOR Loop
echo "<h3>Numbers 1 to 10:</h3>";
for ($i = 1; $i <= 10; $i++) {
    echo $i . " ";
}

// WHILE Loop
echo "<h3>Countdown:</h3>";
$count = 5;
while ($count > 0) {
    echo $count . " ";
    $count--;
}

// FOREACH Loop (for arrays)
$colors = array("Red", "Green", "Blue");
foreach ($colors as $color) {
    echo $color . "<br>";
}
?>
```

MODULE 3: PHP ARRAYS AND FUNCTIONS

3.1 Arrays

```
<?php
// Indexed Array
$fruits = array("Apple", "Banana", "Orange");
echo $fruits[0];  // Apple
echo $fruits[1];  // Banana

// Alternative syntax (PHP 5.4+)
$vegetables = ["Carrot", "Tomato", "Potato"];

// Associative Array (key-value pairs)
$student = array(
    "name" => "John Kamau",
    "age" => 20,
    "course" => "Computer Science",
    "gpa" => 3.7
);

echo $student["name"];    // John Kamau
echo $student["course"];  // Computer Science

// Multidimensional Array
$students = array(
    array("John", 20, "CS"),
    array("Mary", 21, "IT"),
    array("Peter", 19, "Engineering")
);

echo $students[0][0];  // John
echo $students[1][2];  // IT

// Array Functions
$numbers = array(5, 2, 8, 1, 9);

echo "Count: " . count($numbers) . "<br>";
echo "Max: " . max($numbers) . "<br>";
echo "Min: " . min($numbers) . "<br>";

sort($numbers);  // Sort ascending
print_r($numbers);

// Adding to array
array_push($fruits, "Mango");
$fruits[] = "Pineapple";

// Checking if value exists
if (in_array("Apple", $fruits)) {
    echo "Apple found!";
}
?>
```

3.2 Functions

```php
<?php
// Simple Function
function greet() {
    echo "Hello, World!";
}

greet();  // Call function

// Function with Parameters
function greetUser($name) {
    echo "Hello, $name!";
}

greetUser("Alice");  // Hello, Alice!
greetUser("Bob");    // Hello, Bob!

// Function with Return Value
function add($a, $b) {
    return $a + $b;
}

$result = add(5, 3);
echo "Result: $result";  // Result: 8

// Function with Default Parameter
function greetWithTime($name, $time = "morning") {
    return "Good $time, $name!";
}

echo greetWithTime("John");              // Good morning, John!
echo greetWithTime("Mary", "evening");   // Good evening, Mary!

// Function returning multiple values (using array)
function getStudentInfo() {
    return array(
        "name" => "John",
        "age" => 20,
        "gpa" => 3.7
    );
}

$info = getStudentInfo();
echo $info["name"];  // John

// Practical Example: Calculate Grade
function calculateGrade($score) {
    if ($score >= 90) return "A";
    elseif ($score >= 80) return "B";
    elseif ($score >= 70) return "C";
    elseif ($score >= 60) return "D";
    else return "F";
}
```

```php
echo "Score 85 = Grade " . calculateGrade(85);  // Grade B

// Practical Example: Calculate Percentage
function calculatePercentage($obtained, $total) {
    return ($obtained / $total) * 100;
}

$percentage = calculatePercentage(450, 500);
echo "Percentage: " . number_format($percentage, 2) . "%";
?>
```

MODULE 4: PHP AND HTML INTEGRATION

4.1 Embedding PHP in HTML

```
<!DOCTYPE html>
<html>
<head>
    <title>Student List</title>
    <style>
        table { border-collapse: collapse; width: 100%; }
        th, td { border: 1px solid black; padding: 10px; text-align: left; }
        th { background-color: #4CAF50; color: white; }
    </style>
</head>
<body>
    <h1>Student Records</h1>

    <?php
    // Create array of students
    $students = array(
        array("S001", "John Kamau", "Computer Science", 3.7),
        array("S002", "Mary Wanjiru", "Information Technology", 3.9),
        array("S003", "Peter Ochieng", "Engineering", 3.5),
        array("S004", "Grace Akinyi", "Business IT", 3.8)
    );
    ?>

    <table>
        <tr>
            <th>Student ID</th>
            <th>Name</th>
            <th>Course</th>
            <th>GPA</th>
            <th>Status</th>
        </tr>

        <?php foreach ($students as $student): ?>
        <tr>
            <td><?php echo $student[0]; ?></td>
            <td><?php echo $student[1]; ?></td>
            <td><?php echo $student[2]; ?></td>
            <td><?php echo $student[3]; ?></td>
            <td>
                <?php
                if ($student[3] >= 3.5) {
                    echo "<span style='color: green;'>Honors</span>";
                } else {
                    echo "<span style='color: blue;'>Regular</span>";
                }
                ?>
            </td>
        </tr>
        <?php endforeach; ?>
    </table>
```

```
    <h3>Statistics</h3>
    <?php
    $totalGPA = 0;
    foreach ($students as $student) {
        $totalGPA += $student[3];
    }
    $averageGPA = $totalGPA / count($students);
    ?>

    <p>Total Students: <?php echo count($students); ?></p>
    <p>Average GPA: <?php echo number_format($averageGPA, 2); ?></p>
</body>
</html>
```

4.2 Dynamic Content Generation

```
<?php
// Dynamic Navigation Menu
$currentPage = "home";  // This could come from URL parameter
?>

<!DOCTYPE html>
<html>
<head>
    <title>Dynamic Website</title>
    <style>
        nav ul { list-style: none; padding: 0; background: #333; }
        nav li { display: inline-block; }
        nav a {
            display: block;
            padding: 15px 20px;
            color: white;
            text-decoration: none;
        }
        nav a.active, nav a:hover { background: #4CAF50; }
    </style>
</head>
<body>
    <nav>
        <ul>
            <?php
            $menuItems = array(
                "home" => "Home",
                "about" => "About",
                "services" => "Services",
                "contact" => "Contact"
            );

            foreach ($menuItems as $page => $label) {
                $activeClass = ($page == $currentPage) ? 'class="active"' : '';
                echo "<li><a href='$page.php' $activeClass>$label</a></li>";
            }
            ?>
        </ul>
    </nav>
```

```
    <h1>Welcome to <?php echo ucfirst($currentPage); ?></h1>

    <?php
    // Display time-based greeting
    $hour = date("H");
    if ($hour < 12) {
        $greeting = "Good Morning";
    } elseif ($hour < 18) {
        $greeting = "Good Afternoon";
    } else {
        $greeting = "Good Evening";
    }
    ?>

    <p><?php echo $greeting; ?>! Today is <?php echo date("l, F j, Y"); ?></p>
</body>
</html>
```

MODULE 5: FORM HANDLING WITH PHP

5.1 HTML Forms Basics

```
<!-- registration.php -->
<!DOCTYPE html>
<html>
<head>
    <title>Student Registration</title>
    <style>
        form { max-width: 500px; margin: 50px auto; padding: 20px;
               border: 1px solid #ddd; border-radius: 5px; }
        input, select { width: 100%; padding: 10px; margin: 10px 0;
                        border: 1px solid #ddd; border-radius: 3px; }
        button { background: #4CAF50; color: white; padding: 12px 30px;
                 border: none; border-radius: 3px; cursor: pointer; }
        button:hover { background: #45a049; }
        .error { color: red; font-size: 14px; }
    </style>
</head>
<body>
    <h2 style="text-align: center;">Student Registration Form</h2>

    <form method="POST" action="process_registration.php">
        <label>Full Name:</label>
        <input type="text" name="fullname" required>

        <label>Email:</label>
        <input type="email" name="email" required>

        <label>Age:</label>
        <input type="number" name="age" min="16" max="100" required>

        <label>Course:</label>
        <select name="course" required>
            <option value="">-- Select Course --</option>
            <option value="Computer Science">Computer Science</option>
            <option value="Information Technology">Information
Technology</option>
            <option value="Engineering">Engineering</option>
            <option value="Business IT">Business IT</option>
        </select>

        <label>Gender:</label>
        <input type="radio" name="gender" value="Male" required> Male
        <input type="radio" name="gender" value="Female" required> Female

        <button type="submit">Register</button>
    </form>
</body>
</html>
```

5.2 Processing Form Data (POST Method)

```php
<?php
// process_registration.php

if ($_SERVER["REQUEST_METHOD"] == "POST") {

    // Retrieve form data
    $fullname = $_POST["fullname"];
    $email = $_POST["email"];
    $age = $_POST["age"];
    $course = $_POST["course"];
    $gender = $_POST["gender"];

    // Display received data
    echo "<h2>Registration Successful!</h2>";
    echo "<p><strong>Full Name:</strong> $fullname</p>";
    echo "<p><strong>Email:</strong> $email</p>";
    echo "<p><strong>Age:</strong> $age</p>";
    echo "<p><strong>Course:</strong> $course</p>";
    echo "<p><strong>Gender:</strong> $gender</p>";

    echo "<p><a href='registration.php'>Register Another Student</a></p>";

} else {
    echo "Invalid request method";
}
?>
```

5.3 Form Validation

```php
<?php
// validate_registration.php

$errors = array();

if ($_SERVER["REQUEST_METHOD"] == "POST") {

    // Get and sanitize input
    $fullname = trim($_POST["fullname"]);
    $email = trim($_POST["email"]);
    $age = $_POST["age"];
    $course = $_POST["course"];

    // Validation
    if (empty($fullname)) {
        $errors[] = "Full name is required";
    } elseif (strlen($fullname) < 3) {
        $errors[] = "Full name must be at least 3 characters";
    }

    if (empty($email)) {
        $errors[] = "Email is required";
    } elseif (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
        $errors[] = "Invalid email format";
    }
```

```
    if (empty($age)) {
        $errors[] = "Age is required";
    } elseif ($age < 16 || $age > 100) {
        $errors[] = "Age must be between 16 and 100";
    }

    if (empty($course)) {
        $errors[] = "Please select a course";
    }

    // Check if there are errors
    if (empty($errors)) {
        echo "<h2>Registration Successful!</h2>";
        echo "<p>Thank you, $fullname! Your registration has been
submitted.</p>";
        // Here you would normally insert into database
    } else {
        echo "<h2>Registration Errors:</h2>";
        echo "<ul>";
        foreach ($errors as $error) {
            echo "<li style='color:red;'>$error</li>";
        }
        echo "</ul>";
        echo "<p><a href='registration.php'>Go Back</a></p>";
    }
}
?>
```

MODULE 6: MYSQL DATABASE FUNDAMENTALS

6.1 Creating a Database

Using PHPMyAdmin:

1. Open http://localhost/phpmyadmin

2. Click "New" to create database

3. Enter database name: student_portal

4. Click "Create"

Using SQL:

```
-- Create Database
CREATE DATABASE student_portal;

-- Use the database
USE student_portal;

-- Create Students Table
CREATE TABLE students (
    student_id INT AUTO_INCREMENT PRIMARY KEY,
    fullname VARCHAR(100) NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    age INT NOT NULL,
    course VARCHAR(100) NOT NULL,
    gender ENUM('Male', 'Female') NOT NULL,
    registration_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Create Courses Table
CREATE TABLE courses (
    course_id INT AUTO_INCREMENT PRIMARY KEY,
    course_name VARCHAR(100) NOT NULL,
    course_code VARCHAR(10) UNIQUE NOT NULL,
    credits INT NOT NULL
);

-- Insert Sample Courses
INSERT INTO courses (course_name, course_code, credits) VALUES
('Computer Science', 'CS101', 3),
('Information Technology', 'IT101', 3),
('Software Engineering', 'SE101', 4),
('Data Science', 'DS101', 4);

-- Create Users Table (for authentication)
CREATE TABLE users (
    user_id INT AUTO_INCREMENT PRIMARY KEY,
```

```
    username VARCHAR(50) UNIQUE NOT NULL,
    password VARCHAR(255) NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    role ENUM('admin', 'student') DEFAULT 'student',
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

MODULE 7: PHP-MYSQL CONNECTION

7.1 Database Connection File

```
<?php
// config/database.php

// Database configuration
define('DB_HOST', 'localhost');
define('DB_USER', 'root');
define('DB_PASS', '');  // Default XAMPP password is empty
define('DB_NAME', 'student_portal');

// Create connection
$conn = mysqli_connect(DB_HOST, DB_USER, DB_PASS, DB_NAME);

// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}

// Set charset to UTF-8
mysqli_set_charset($conn, "utf8");

// Uncomment to display success message
// echo "Connected successfully";
?>
```

7.2 Using PDO (Better Approach)

```
<?php
// config/database_pdo.php

$host = 'localhost';
$dbname = 'student_portal';
$username = 'root';
$password = '';

try {
    // Create PDO connection
    $pdo = new PDO(
        "mysql:host=$host;dbname=$dbname;charset=utf8",
        $username,
        $password
    );

    // Set error mode to exceptions
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

    // Uncomment to display success message
    // echo "Connected successfully using PDO";
```

```
} catch (PDOException $e) {
    die("Connection failed: " . $e->getMessage());
}
?>
```

MODULE 8: CRUD OPERATIONS

8.1 CREATE - Insert Data

```
<?php
// add_student.php
include 'config/database.php';

if ($_SERVER["REQUEST_METHOD"] == "POST") {

    $fullname = mysqli_real_escape_string($conn, $_POST['fullname']);
    $email = mysqli_real_escape_string($conn, $_POST['email']);
    $age = $_POST['age'];
    $course = mysqli_real_escape_string($conn, $_POST['course']);
    $gender = $_POST['gender'];

    $sql = "INSERT INTO students (fullname, email, age, course, gender)
            VALUES ('$fullname', '$email', '$age', '$course', '$gender')";

    if (mysqli_query($conn, $sql)) {
        echo "Student added successfully!";
        $student_id = mysqli_insert_id($conn);
        echo " Student ID: $student_id";
    } else {
        echo "Error: " . mysqli_error($conn);
    }
}

mysqli_close($conn);
?>

<!-- HTML Form -->
<form method="POST">
    <input type="text" name="fullname" placeholder="Full Name" required><br>
    <input type="email" name="email" placeholder="Email" required><br>
    <input type="number" name="age" placeholder="Age" required><br>
    <select name="course" required>
        <option value="Computer Science">Computer Science</option>
        <option value="IT">Information Technology</option>
    </select><br>
    <input type="radio" name="gender" value="Male" required> Male
    <input type="radio" name="gender" value="Female" required> Female<br>
    <button type="submit">Add Student</button>
</form>
```

8.2 READ - Display Data

```
<?php
// view_students.php
include 'config/database.php';

$sql = "SELECT * FROM students ORDER BY student_id DESC";
$result = mysqli_query($conn, $sql);
?>

<!DOCTYPE html>
<html>
```

```
<head>
    <title>All Students</title>
    <style>
        table { width: 100%; border-collapse: collapse; }
        th, td { border: 1px solid #ddd; padding: 12px; text-align: left; }
        th { background-color: #4CAF50; color: white; }
        tr:hover { background-color: #f5f5f5; }
        .actions a { padding: 5px 10px; margin: 0 5px; text-decoration: none; }
        .edit { background: #2196F3; color: white; }
        .delete { background: #f44336; color: white; }
    </style>
</head>
<body>
    <h2>All Students</h2>

    <?php if (mysqli_num_rows($result) > 0): ?>
    <table>
        <tr>
            <th>ID</th>
            <th>Full Name</th>
            <th>Email</th>
            <th>Age</th>
            <th>Course</th>
            <th>Gender</th>
            <th>Registration Date</th>
            <th>Actions</th>
        </tr>

        <?php while($row = mysqli_fetch_assoc($result)): ?>
        <tr>
            <td><?php echo $row['student_id']; ?></td>
            <td><?php echo $row['fullname']; ?></td>
            <td><?php echo $row['email']; ?></td>
            <td><?php echo $row['age']; ?></td>
            <td><?php echo $row['course']; ?></td>
            <td><?php echo $row['gender']; ?></td>
            <td><?php echo date('M d, Y', strtotime($row['registration_date'])); ?></td>
            <td class="actions">
                <a href="edit_student.php?id=<?php echo $row['student_id']; ?>"
                   class="edit">Edit</a>
                <a href="delete_student.php?id=<?php echo $row['student_id']; ?>"
                   class="delete"
                   onclick="return confirm('Delete this student?')">Delete</a>
            </td>
        </tr>
        <?php endwhile; ?>
    </table>

    <p>Total Students: <?php echo mysqli_num_rows($result); ?></p>

    <?php else: ?>
        <p>No students found.</p>
    <?php endif; ?>

    <?php mysqli_close($conn); ?>
</body>
</html>
```

8.3 UPDATE - Edit Data

```
<?php
// edit_student.php
include 'config/database.php';

// Get student ID from URL
$student_id = $_GET['id'];

// Fetch student data
$sql = "SELECT * FROM students WHERE student_id = $student_id";
$result = mysqli_query($conn, $sql);
$student = mysqli_fetch_assoc($result);

// Handle form submission
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $fullname = mysqli_real_escape_string($conn, $_POST['fullname']);
    $email = mysqli_real_escape_string($conn, $_POST['email']);
    $age = $_POST['age'];
    $course = mysqli_real_escape_string($conn, $_POST['course']);
    $gender = $_POST['gender'];

    $sql = "UPDATE students SET
            fullname = '$fullname',
            email = '$email',
            age = '$age',
            course = '$course',
            gender = '$gender'
            WHERE student_id = $student_id";

    if (mysqli_query($conn, $sql)) {
        echo "Student updated successfully!";
        header("Location: view_students.php");
    } else {
        echo "Error: " . mysqli_error($conn);
    }
}
?>

<form method="POST">
    <input type="text" name="fullname" value="<?php echo $student['fullname'];
?>" required><br>
    <input type="email" name="email" value="<?php echo $student['email']; ?>"
required><br>
    <input type="number" name="age" value="<?php echo $student['age']; ?>"
required><br>
    <select name="course" required>
        <option value="Computer Science"
            <?php echo ($student['course'] == 'Computer Science') ? 'selected'
: ''; ?>>
            Computer Science</option>
        <option value="IT"
            <?php echo ($student['course'] == 'IT') ? 'selected' : ''; ?>>
            Information Technology</option>
    </select><br>
```

```
    <input type="radio" name="gender" value="Male"
        <?php echo ($student['gender'] == 'Male') ? 'checked' : ''; ?>> Male
    <input type="radio" name="gender" value="Female"
        <?php echo ($student['gender'] == 'Female') ? 'checked' : ''; ?>>
Female<br>
    <button type="submit">Update Student</button>
    <a href="view_students.php">Cancel</a>
</form>
```

8.4 DELETE - Remove Data

```
<?php
// delete_student.php
include 'config/database.php';

if (isset($_GET['id'])) {
    $student_id = $_GET['id'];

    $sql = "DELETE FROM students WHERE student_id = $student_id";

    if (mysqli_query($conn, $sql)) {
        echo "Student deleted successfully!";
        header("Location: view_students.php");
    } else {
        echo "Error: " . mysqli_error($conn);
    }
}

mysqli_close($conn);
?>
```

MODULE 9: SESSIONS AND COOKIES

9.1 Sessions

Sessions store user data on the server.

```
<?php
// login.php - Setting session
session_start();

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $username = $_POST['username'];
    $password = $_POST['password'];

    // Verify credentials (simplified)
    if ($username == "admin" && $password == "password") {
        // Set session variables
        $_SESSION['logged_in'] = true;
        $_SESSION['username'] = $username;
        $_SESSION['user_id'] = 1;
        $_SESSION['role'] = 'admin';

        header("Location: dashboard.php");
    } else {
        echo "Invalid credentials";
    }
}
?>

<form method="POST">
    <input type="text" name="username" placeholder="Username" required><br>
    <input type="password" name="password" placeholder="Password" required><br>
    <button type="submit">Login</button>
</form>
```

```
<?php
// dashboard.php - Reading session
session_start();

// Check if user is logged in
if (!isset($_SESSION['logged_in']) || $_SESSION['logged_in'] !== true)
{
    header("Location: login.php");
    exit;
}

echo "<h2>Welcome, " . $_SESSION['username'] . "!</h2>";
echo "<p>User ID: " . $_SESSION['user_id'] . "</p>";
echo "<p>Role: " . $_SESSION['role'] . "</p>";
echo "<p><a href='logout.php'>Logout</a></p>";
?>
```

```php
<?php
// logout.php - Destroying session
session_start();
session_destroy();
header("Location: login.php");
?>
```

9.2 Cookies

Cookies store data on the client browser.

```php
<?php
// Setting a cookie (valid for 30 days)
setcookie("username", "JohnDoe", time() + (86400 * 30), "/");
setcookie("theme", "dark", time() + (86400 * 30), "/");

// Reading a cookie
if (isset($_COOKIE["username"])) {
    echo "Welcome back, " . $_COOKIE["username"];
} else {
    echo "Welcome, new visitor!";
}

// Deleting a cookie (set expiry to past time)
setcookie("username", "", time() - 3600, "/");
?>
```

MODULE 10: USER AUTHENTICATION SYSTEM

10.1 Complete Registration System

```
<?php
// register.php
include 'config/database.php';

$errors = array();

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $username = mysqli_real_escape_string($conn, $_POST['username']);
    $email = mysqli_real_escape_string($conn, $_POST['email']);
    $password = $_POST['password'];
    $confirm_password = $_POST['confirm_password'];

    // Validation
    if (empty($username) || empty($email) || empty($password)) {
        $errors[] = "All fields are required";
    }

    if ($password !== $confirm_password) {
        $errors[] = "Passwords do not match";
    }

    if (strlen($password) < 6) {
        $errors[] = "Password must be at least 6 characters";
    }

    // Check if username exists
    $check = "SELECT * FROM users WHERE username = '$username'";
    if (mysqli_num_rows(mysqli_query($conn, $check)) > 0) {
        $errors[] = "Username already exists";
    }

    // Check if email exists
    $check = "SELECT * FROM users WHERE email = '$email'";
    if (mysqli_num_rows(mysqli_query($conn, $check)) > 0) {
        $errors[] = "Email already registered";
    }

    if (empty($errors)) {
        // Hash password
        $hashed_password = password_hash($password, PASSWORD_DEFAULT);

        $sql = "INSERT INTO users (username, email, password)
                VALUES ('$username', '$email', '$hashed_password')";

        if (mysqli_query($conn, $sql)) {
            echo "Registration successful! <a href='login.php'>Login here</a>";
        } else {
            echo "Error: " . mysqli_error($conn);
        }
    } else {
        foreach ($errors as $error) {
```

```
            echo "<p style='color:red;'>$error</p>";
        }
    }
}
?>

<form method="POST">
    <input type="text" name="username" placeholder="Username" required><br>
    <input type="email" name="email" placeholder="Email" required><br>
    <input type="password" name="password" placeholder="Password" required><br>
    <input type="password" name="confirm_password" placeholder="Confirm
Password" required><br>
    <button type="submit">Register</button>
</form>
```

10.2 Complete Login System

```
<?php
// login.php
session_start();
include 'config/database.php';

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $username = mysqli_real_escape_string($conn, $_POST['username']);
    $password = $_POST['password'];

    $sql = "SELECT * FROM users WHERE username = '$username'";
    $result = mysqli_query($conn, $sql);

    if (mysqli_num_rows($result) == 1) {
        $user = mysqli_fetch_assoc($result);

        // Verify password
        if (password_verify($password, $user['password'])) {
            // Set session variables
            $_SESSION['logged_in'] = true;
            $_SESSION['user_id'] = $user['user_id'];
            $_SESSION['username'] = $user['username'];
            $_SESSION['role'] = $user['role'];

            // Redirect based on role
            if ($user['role'] == 'admin') {
                header("Location: admin/dashboard.php");
            } else {
                header("Location: student/dashboard.php");
            }
        } else {
            $error = "Invalid password";
        }
    } else {
        $error = "User not found";
    }
}
?>
```

```
<!DOCTYPE html>
<html>
<head>
    <title>Login</title>
    <style>
        .login-container { max-width: 400px; margin: 100px auto;
                           padding: 30px; border: 1px solid #ddd; }
        input { width: 100%; padding: 10px; margin: 10px 0; }
        button { width: 100%; padding: 12px; background: #4CAF50;
                 color: white; border: none; cursor: pointer; }
        .error { color: red; text-align: center; }
    </style>
</head>
<body>
    <div class="login-container">
        <h2>Login</h2>

        <?php if (isset($error)): ?>
            <p class="error"><?php echo $error; ?></p>
        <?php endif; ?>

        <form method="POST">
            <input type="text" name="username" placeholder="Username" required>
            <input type="password" name="password" placeholder="Password"
required>
            <button type="submit">Login</button>
        </form>

        <p>Don't have an account? <a href="register.php">Register here</a></p>
    </div>
</body>
</html>
```

10.3 Authentication Middleware

```
<?php
// middleware/auth.php

function requireLogin() {
    if (!isset($_SESSION['logged_in']) || $_SESSION['logged_in'] !==
true) {
        header("Location: ../login.php");
        exit;
    }
}

function requireAdmin() {
    requireLogin();
    if ($_SESSION['role'] !== 'admin') {
        die("Access denied. Admin only.");
    }
}
```

```
// Usage in protected pages:
// session_start();
// include '../middleware/auth.php';
// requireLogin();  // or requireAdmin();
?>
```

MODULE 11: FILE UPLOAD AND MANAGEMENT

```
<?php
// upload.php

if ($_SERVER["REQUEST_METHOD"] == "POST") {

    $target_dir = "uploads/";
    $file = $_FILES["file"];
    $filename = basename($file["name"]);
    $target_file = $target_dir . time() . "_" . $filename;

    // File validation
    $allowed_types = array("jpg", "jpeg", "png", "pdf");
    $file_extension = strtolower(pathinfo($target_file, PATHINFO_EXTENSION));

    if (!in_array($file_extension, $allowed_types)) {
        die("Invalid file type. Allowed: JPG, JPEG, PNG, PDF");
    }

    if ($file["size"] > 5000000) {  // 5MB max
        die("File too large. Maximum 5MB");
    }

    if (move_uploaded_file($file["tmp_name"], $target_file)) {
        echo "File uploaded successfully: " . $target_file;

        // Save to database
        include 'config/database.php';
        $sql = "INSERT INTO uploads (filename, filepath, uploaded_by)
                VALUES ('$filename', '$target_file',
'{$_SESSION['user_id']}')";
        mysqli_query($conn, $sql);

    } else {
        echo "Error uploading file";
    }
}
?>

<form method="POST" enctype="multipart/form-data">
    <input type="file" name="file" required>
    <button type="submit">Upload</button>
</form>
```

MODULE 12: JAVASCRIPT FOR WEB APPLICATIONS

12.1 Client-Side Form Validation

```
<!DOCTYPE html>
<html>
<head>
    <title>Form Validation</title>
</head>
<body>
    <form id="studentForm" onsubmit="return validateForm()">
        <input type="text" id="name" placeholder="Full Name">
        <span id="nameError" style="color:red;"></span><br>

        <input type="email" id="email" placeholder="Email">
        <span id="emailError" style="color:red;"></span><br>

        <input type="number" id="age" placeholder="Age">
        <span id="ageError" style="color:red;"></span><br>

        <button type="submit">Submit</button>
    </form>

    <script>
    function validateForm() {
        // Clear previous errors
        document.getElementById('nameError').innerHTML = '';
        document.getElementById('emailError').innerHTML = '';
        document.getElementById('ageError').innerHTML = '';

        let isValid = true;

        // Name validation
        let name = document.getElementById('name').value;
        if (name.length < 3) {
            document.getElementById('nameError').innerHTML =
                'Name must be at least 3 characters';
            isValid = false;
        }

        // Email validation
        let email = document.getElementById('email').value;
        let emailPattern = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
        if (!emailPattern.test(email)) {
            document.getElementById('emailError').innerHTML =
                'Invalid email format';
            isValid = false;
        }

        // Age validation
        let age = document.getElementById('age').value;
        if (age < 16 || age > 100) {
            document.getElementById('ageError').innerHTML =
                'Age must be between 16 and 100';
            isValid = false;
```

```
        }

        return isValid;
    }
    </script>
</body>
</html>
```

MODULE 13: AJAX - ASYNCHRONOUS JAVASCRIPT AND XML

13.1 AJAX with Pure JavaScript

```
<!-- index.html -->
<!DOCTYPE html>
<html>
<head>
    <title>AJAX Example</title>
</head>
<body>
    <h2>Search Students</h2>
    <input type="text" id="searchBox" onkeyup="searchStudents()"
           placeholder="Type to search...">
    <div id="results"></div>

    <script>
    function searchStudents() {
        let query = document.getElementById('searchBox').value;

        if (query.length == 0) {
            document.getElementById('results').innerHTML = '';
            return;
        }

        // Create XMLHttpRequest object
        let xhr = new XMLHttpRequest();

        // Define callback function
        xhr.onreadystatechange = function() {
            if (this.readyState == 4 && this.status == 200) {
                document.getElementById('results').innerHTML =
this.responseText;
            }
        };

        // Send request
        xhr.open('GET', 'search.php?q=' + query, true);
        xhr.send();
    }
    </script>
</body>
</html>

<?php
// search.php
include 'config/database.php';

$query = mysqli_real_escape_string($conn, $_GET['q']);

$sql = "SELECT * FROM students WHERE fullname LIKE '%$query%'
        OR email LIKE '%$query%' LIMIT 10";
$result = mysqli_query($conn, $sql);

if (mysqli_num_rows($result) > 0) {
```

```
    echo "<table border='1'>";
    echo "<tr><th>Name</th><th>Email</th><th>Course</th></tr>";

    while ($row = mysqli_fetch_assoc($result)) {
        echo "<tr>";
        echo "<td>" . $row['fullname'] . "</td>";
        echo "<td>" . $row['email'] . "</td>";
        echo "<td>" . $row['course'] . "</td>";
        echo "</tr>";
    }

    echo "</table>";
} else {
    echo "No results found";
}
?>
```

13.2 AJAX with jQuery (Simpler)

```
<!DOCTYPE html>
<html>
<head>
    <title>AJAX with jQuery</title>
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
</head>
<body>
    <input type="text" id="searchBox" placeholder="Search...">
    <div id="results"></div>

    <script>
    $(document).ready(function() {
        $('#searchBox').keyup(function() {
            let query = $(this).val();

            if (query != '') {
                $.ajax({
                    url: 'search.php',
                    method: 'GET',
                    data: {q: query},
                    success: function(data) {
                        $('#results').html(data);
                    }
                });
            } else {
                $('#results').html('');
            }
        });
    });
    </script>
</body>
</html>
```

13.3 AJAX with Fetch API (Modern)

```
<script>
async function searchStudents() {
```

```
    let query = document.getElementById('searchBox').value;

    if (query.length == 0) {
        document.getElementById('results').innerHTML = '';
        return;
    }

    try {
        let response = await fetch('search.php?q=' + query);
        let data = await response.text();
        document.getElementById('results').innerHTML = data;
    } catch (error) {
        console.error('Error:', error);
    }
}
</script>
```

MODULE 14: REST API DEVELOPMENT

14.1 Creating a Simple REST API

```
<?php
// api/students.php
header('Content-Type: application/json');
include '../config/database.php';

$method = $_SERVER['REQUEST_METHOD'];

switch ($method) {
    case 'GET':
        // Get all students or specific student
        if (isset($_GET['id'])) {
            $id = $_GET['id'];
            $sql = "SELECT * FROM students WHERE student_id = $id";
        } else {
            $sql = "SELECT * FROM students";
        }

        $result = mysqli_query($conn, $sql);
        $students = array();

        while ($row = mysqli_fetch_assoc($result)) {
            $students[] = $row;
        }

        echo json_encode([
            'success' => true,
            'data' => $students
        ]);
        break;

    case 'POST':
        // Create new student
        $data = json_decode(file_get_contents('php://input'), true);

        $fullname = $data['fullname'];
        $email = $data['email'];
        $age = $data['age'];
        $course = $data['course'];
        $gender = $data['gender'];

        $sql = "INSERT INTO students (fullname, email, age, course, gender)
                VALUES ('$fullname', '$email', '$age', '$course', '$gender')";

        if (mysqli_query($conn, $sql)) {
            echo json_encode([
                'success' => true,
                'message' => 'Student created',
                'id' => mysqli_insert_id($conn)
            ]);
        } else {
            echo json_encode([
                'success' => false,
                'message' => mysqli_error($conn)
            ]);
        }
        break;
```

```
    case 'PUT':
        // Update student
        $data = json_decode(file_get_contents('php://input'), true);
        $id = $data['id'];

        $sql = "UPDATE students SET
                fullname = '{$data['fullname']}',
                email = '{$data['email']}',
                age = '{$data['age']}',
                course = '{$data['course']}'
                WHERE student_id = $id";

        if (mysqli_query($conn, $sql)) {
            echo json_encode(['success' => true, 'message' => 'Student updated']);
        } else {
            echo json_encode(['success' => false, 'message' => mysqli_error($conn)]);
        }
        break;

    case 'DELETE':
        // Delete student
        parse_str(file_get_contents('php://input'), $data);
        $id = $data['id'];

        $sql = "DELETE FROM students WHERE student_id = $id";

        if (mysqli_query($conn, $sql)) {
            echo json_encode(['success' => true, 'message' => 'Student deleted']);
        } else {
            echo json_encode(['success' => false, 'message' => mysqli_error($conn)]);
        }
        break;

    default:
        echo json_encode(['success' => false, 'message' => 'Invalid method']);
}
?>
```

14.2 Consuming the API with JavaScript

```
<script>
// GET - Fetch all students
async function getStudents() {
    let response = await fetch('api/students.php');
    let data = await response.json();
    console.log(data.data);
}

// POST - Create new student
async function createStudent() {
    let student = {
        fullname: 'Alice Muthoni',
        email: 'alice@example.com',
        age: 20,
        course: 'Computer Science',
        gender: 'Female'
    };

    let response = await fetch('api/students.php', {
        method: 'POST',
```

```
        headers: {'Content-Type': 'application/json'},
        body: JSON.stringify(student)
    });

    let data = await response.json();
    console.log(data);
}

// PUT - Update student
async function updateStudent(id) {
    let student = {
        id: id,
        fullname: 'Alice Njeri',
        email: 'alice.new@example.com',
        age: 21,
        course: 'IT'
    };

    let response = await fetch('api/students.php', {
        method: 'PUT',
        body: JSON.stringify(student)
    });

    let data = await response.json();
    console.log(data);
}

// DELETE - Delete student
async function deleteStudent(id) {
    let response = await fetch('api/students.php', {
        method: 'DELETE',
        body: 'id=' + id
    });

    let data = await response.json();
    console.log(data);
}
</script>
```

MODULE 15: SECURITY BEST PRACTICES

15.1 SQL Injection Prevention

```
<?php
// BAD - Vulnerable to SQL Injection
$username = $_POST['username'];
$sql = "SELECT * FROM users WHERE username = '$username'";  // DANGEROUS!

// GOOD - Using Prepared Statements (mysqli)
$username = $_POST['username'];
$stmt = $conn->prepare("SELECT * FROM users WHERE username = ?");
$stmt->bind_param("s", $username);
$stmt->execute();
$result = $stmt->get_result();

// GOOD - Using Prepared Statements (PDO)
$username = $_POST['username'];
$stmt = $pdo->prepare("SELECT * FROM users WHERE username = :username");
$stmt->execute(['username' => $username]);
$user = $stmt->fetch();
?>
```

15.2 XSS (Cross-Site Scripting) Prevention

```
<?php
// BAD - Vulnerable to XSS
echo $_GET['name'];  // User can inject scripts

// GOOD - Sanitize output
echo htmlspecialchars($_GET['name'], ENT_QUOTES, 'UTF-8');

// Function for safe output
function safe_output($text) {
    return htmlspecialchars($text, ENT_QUOTES, 'UTF-8');
}

echo safe_output($user_input);
?>
```

15.3 CSRF (Cross-Site Request Forgery) Prevention

```
<?php
// Generate CSRF token
session_start();
if (empty($_SESSION['csrf_token'])) {
    $_SESSION['csrf_token'] = bin2hex(random_bytes(32));
}
?>

<!-- Add to form -->
<form method="POST">
    <input type="hidden" name="csrf_token" value="<?php echo
$_SESSION['csrf_token']; ?>">
```

```
    <!-- Other form fields -->
</form>

<?php
// Verify CSRF token on submission
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    if (!isset($_POST['csrf_token']) || $_POST['csrf_token'] !==
$_SESSION['csrf_token']) {
        die('Invalid CSRF token');
    }
    // Process form
}
?>
```

15.4 Security Checklist

Essential Security Practices:

- Always use prepared statements for database queries
- Validate and sanitize all user inputs
- Use password_hash() for storing passwords
- Implement CSRF tokens for forms
- Use HTTPS in production
- Set secure session cookies
- Limit file upload types and sizes
- Keep PHP and libraries updated
- Use environment variables for sensitive data
- Implement rate limiting for login attempts

MODULE 16: EMAIL INTEGRATION WITH PHPMAILER

```
<?php
// Install PHPMailer: composer require phpmailer/phpmailer
// Or download from: github.com/PHPMailer/PHPMailer

use PHPMailer\PHPMailer\PHPMailer;
use PHPMailer\PHPMailer\Exception;

require 'vendor/autoload.php';

function sendEmail($to, $subject, $body) {
    $mail = new PHPMailer(true);

    try {
        // Server settings
        $mail->isSMTP();
        $mail->Host = 'smtp.gmail.com';
        $mail->SMTPAuth = true;
        $mail->Username = 'your_email@gmail.com';
        $mail->Password = 'your_app_password';  // Use App Password
        $mail->SMTPSecure = PHPMailer::ENCRYPTION_STARTTLS;
        $mail->Port = 587;

        // Recipients
        $mail->setFrom('your_email@gmail.com', 'Student Portal');
        $mail->addAddress($to);

        // Content
        $mail->isHTML(true);
        $mail->Subject = $subject;
        $mail->Body = $body;

        $mail->send();
        return true;
    } catch (Exception $e) {
        return false;
    }
}

// Usage - Send registration confirmation
$email = 'student@example.com';
$subject = 'Welcome to Student Portal!';
$message = '
<html>
<body>
    <h2>Welcome!</h2>
    <p>Thank you for registering at Student Portal.</p>
    <p>Your account has been created successfully.</p>
    <p>Best regards,<br>Student Portal Team</p>
</body>
</html>';

if (sendEmail($email, $subject, $message)) {
    echo "Email sent successfully";
```

```
} else {
    echo "Email sending failed";
}
?>
```

MODULE 17: PAYMENT GATEWAY INTEGRATION (M-PESA)

17.1 M-Pesa STK Push (Lipa Na M-Pesa)

```
<?php
// mpesa_config.php
define('MPESA_CONSUMER_KEY', 'your_consumer_key');
define('MPESA_CONSUMER_SECRET', 'your_consumer_secret');
define('MPESA_SHORTCODE', 'your_shortcode');
define('MPESA_PASSKEY', 'your_passkey');
define('MPESA_CALLBACK_URL', 'https://yourdomain.com/mpesa_callback.php');

// Get Access Token
function getMpesaAccessToken() {
    $url =
'https://sandbox.safaricom.co.ke/oauth/v1/generate?grant_type=client_credentials';

    $curl = curl_init();
    curl_setopt($curl, CURLOPT_URL, $url);
    $credentials = base64_encode(MPESA_CONSUMER_KEY . ':' . MPESA_CONSUMER_SECRET);
    curl_setopt($curl, CURLOPT_HTTPHEADER, array('Authorization: Basic ' .
$credentials));
    curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);
    curl_setopt($curl, CURLOPT_SSL_VERIFYPEER, false);

    $response = curl_exec($curl);
    $result = json_decode($response);

    return $result->access_token;
}

// Initiate STK Push
function initiateSTKPush($phone, $amount, $accountReference) {
    $accessToken = getMpesaAccessToken();

    $url = 'https://sandbox.safaricom.co.ke/mpesa/stkpush/v1/processrequest';
    $timestamp = date('YmdHis');
    $password = base64_encode(MPESA_SHORTCODE . MPESA_PASSKEY . $timestamp);

    $curl_post_data = array(
        'BusinessShortCode' => MPESA_SHORTCODE,
        'Password' => $password,
        'Timestamp' => $timestamp,
        'TransactionType' => 'CustomerPayBillOnline',
        'Amount' => $amount,
        'PartyA' => $phone,
        'PartyB' => MPESA_SHORTCODE,
        'PhoneNumber' => $phone,
        'CallBackURL' => MPESA_CALLBACK_URL,
        'AccountReference' => $accountReference,
        'TransactionDesc' => 'Payment for ' . $accountReference
    );

    $data_string = json_encode($curl_post_data);

    $curl = curl_init();
    curl_setopt($curl, CURLOPT_URL, $url);
    curl_setopt($curl, CURLOPT_HTTPHEADER, array(
        'Content-Type:application/json',
        'Authorization:Bearer ' . $accessToken
    ));
```

```
    curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);
    curl_setopt($curl, CURLOPT_POST, true);
    curl_setopt($curl, CURLOPT_POSTFIELDS, $data_string);

    $response = curl_exec($curl);
    return json_decode($response);
}

// Usage
$phone = '254700000000';  // Format: 2547XXXXXXXX
$amount = 100;
$reference = 'INV-001';

$result = initiateSTKPush($phone, $amount, $reference);

if ($result->ResponseCode == 0) {
    echo "STK Push sent. Check your phone.";
} else {
    echo "Error: " . $result->ResponseDescription;
}
?>
```

17.2 M-Pesa Callback Handler

```
<?php
// mpesa_callback.php
include 'config/database.php';

$response = file_get_contents('php://input');
$result = json_decode($response, true);

// Log the response
file_put_contents('mpesa_logs.txt', $response . PHP_EOL, FILE_APPEND);

if (isset($result['Body']['stkCallback'])) {
    $callback = $result['Body']['stkCallback'];

    if ($callback['ResultCode'] == 0) {
        // Payment successful
        $items = $callback['CallbackMetadata']['Item'];

        $amount = '';
        $mpesaCode = '';
        $phone = '';

        foreach ($items as $item) {
            if ($item['Name'] == 'Amount') $amount = $item['Value'];
            if ($item['Name'] == 'MpesaReceiptNumber') $mpesaCode =
$item['Value'];
            if ($item['Name'] == 'PhoneNumber') $phone = $item['Value'];
        }

        // Save to database
        $sql = "INSERT INTO payments (phone, amount, mpesa_code, status)
                VALUES ('$phone', '$amount', '$mpesaCode', 'completed')";
        mysqli_query($conn, $sql);

    } else {
```

```
        // Payment failed
        $message = $callback['ResultDesc'];
        // Handle failed payment
    }
}
?>
```

MODULE 18: COMPLETE PROJECT - STUDENT PORTAL

18.1 Project Overview

Features:

- Student registration and login
- Admin dashboard
- Course enrollment
- Grade management
- Fee payment with M-Pesa
- Downloadable reports
- Email notifications

18.2 Database Structure (Simplified)

```
CREATE TABLE students (
    student_id INT AUTO_INCREMENT PRIMARY KEY,
    reg_number VARCHAR(20) UNIQUE,
    fullname VARCHAR(100),
    email VARCHAR(100) UNIQUE,
    phone VARCHAR(15),
    course_id INT,
    year_of_study INT,
    password VARCHAR(255),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE courses (
    course_id INT AUTO_INCREMENT PRIMARY KEY,
    course_name VARCHAR(100),
    course_code VARCHAR(10),
    fee_amount DECIMAL(10,2)
);

CREATE TABLE grades (
    grade_id INT AUTO_INCREMENT PRIMARY KEY,
    student_id INT,
    course_id INT,
    marks DECIMAL(5,2),
    grade VARCHAR(2),
    semester VARCHAR(20),
    FOREIGN KEY (student_id) REFERENCES students(student_id),
    FOREIGN KEY (course_id) REFERENCES courses(course_id)
);

CREATE TABLE payments (
    payment_id INT AUTO_INCREMENT PRIMARY KEY,
    student_id INT,
    amount DECIMAL(10,2),
    mpesa_code VARCHAR(20),
    payment_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
```

```
    status ENUM('pending', 'completed', 'failed'),
    FOREIGN KEY (student_id) REFERENCES students(student_id)
);
```

18.3 Key Implementation Files

Project Structure:

```
student_portal/
├── config/
│   └── database.php
├── admin/
│   ├── dashboard.php
│   ├── manage_students.php
│   └── manage_grades.php
├── student/
│   ├── dashboard.php
│   ├── view_grades.php
│   └── pay_fees.php
├── api/
│   └── students.php
├── login.php
├── register.php
└── logout.php
```

MODULE 19: COMPLETE PROJECT - E-COMMERCE SYSTEM

19.1 E-commerce Features

- Product catalog with categories
- Shopping cart functionality
- User authentication
- Order management
- M-Pesa payment integration
- Admin panel
- Order tracking
- Email notifications

19.2 Key Components

1. Product Listing with AJAX search

2. Add to Cart (using Sessions)

3. Checkout Process

4. Payment Integration

5. Order Confirmation via Email

MODULE 20: DEPLOYMENT AND HOSTING

20.1 Preparing for Deployment

Pre-deployment Checklist:

- Test all features locally
- Remove debug code and error displays
- Update database credentials
- Set error_reporting to production mode
- Enable HTTPS
- Optimize database queries
- Compress images and assets
- Set proper file permissions
- Create database backup
- Test security measures

20.2 Hosting Options in Kenya

- **Truehost Kenya -** truehost.co.ke - Starting KSh 99/month
- **Safaricom Cloud -** cloud.safaricom.co.ke - Enterprise hosting
- **Kenya Web Experts -** kenyawebexperts.com - Local support
- **HostPinnacle -** hostpinnacle.co.ke - Affordable shared hosting
- **SasaHost -** sasahost.co.ke - Domain + Hosting packages

20.3 Deployment Steps (cPanel)

1. Purchase hosting and domain
2. Access cPanel (yourdomain.com/cpanel)
3. Create MySQL database and user
4. Upload files via File Manager or FTP
5. Import database via phpMyAdmin
6. Update config/database.php with new credentials
7. Test website functionality
8. Configure email accounts
9. Set up SSL certificate (Let's Encrypt - Free)
10. Configure domain DNS if needed

CONGRATULATIONS! 🎉

You have completed the comprehensive Internet Applications Programming course!

What You've Learned:

✓ PHP programming fundamentals

✓ MySQL database design and queries

✓ HTML form handling and validation

✓ Sessions and cookies management

✓ User authentication systems

✓ File upload functionality

✓ JavaScript and AJAX

✓ REST API development

✓ Security best practices

✓ Email integration

✓ M-Pesa payment integration

✓ Complete web applications

✓ Deployment strategies

Practice Projects:

Build these to master web development:

1. Library Management System
2. Hospital Appointment System
3. Online Exam System
4. Real Estate Listing Platform
5. Job Portal
6. Restaurant Ordering System

7. Inventory Management

8. Blog/CMS Platform

9. Event Management System

10. Social Media Platform

Additional Resources:

- **PHP Documentation:** php.net/manual
- **MySQL Documentation:** dev.mysql.com/doc
- **W3Schools:** w3schools.com
- **MDN Web Docs:** developer.mozilla.org
- **PHP The Right Way:** phptherightway.com
- **Stack Overflow:** stackoverflow.com
- **GitHub:** github.com - for code examples

Happy Coding! Build Amazing Web Applications! 🚀💻